ANALISIS PERFORMANSI LAYANAN KLUSTER SERVER MENGGUNAKAN PENYEIMBANG BEBAN DAN *VIRTUALBOX*

Fajar Zuhroni[1)], Adian Fatchur Rochim[2)] , Eko Didik Widianto [2)]
Program Studi Sistem Komputer, Fakultas Teknik, Universitas Diponegoro,
Jln. Prof. Sudharto, Tembalang, Semarang, Indonesia
email : zuhroni.f@gmail.com

***Abstract*—,Today, the growth of technology is very fast causes people more familiar with the Internet. Web servers are required to more quickly in order to serve a number of user request. Technology uses a single server is underprivileged to handle the load on the web server traffic as the number of users increases, causing the server to fail in serving the request. Technology on cluster server with load balancer is used in order to divide the load evenly so that optimize performance on a web server.**

Server cluster systems that are designed using six virtual machines using VirtualBox application. Six virtual machine consists of two load balancer servers, two application servers and two database servers. Definition of the system created by outlining system requirements and network topology. System design describes requirements specification of the hardware and software. Analysis and testing conducted to determine the performance of the system designed. Analysis and testing conducted to determine the performance of the system designed.

Results of this research is the design of virtual servers that can serve a number of user request. Test result showed maximum ability to serve when all servers are up reach 240 connection, one of the aplication server down is 180 connection and one of the database down is 220.The optimal result when all servers up is 180 connections, one of the aplication server down is 150 connections and when database server down is 160 connections

Keywords : Server, Virtual, Load Balancing, Web Server

I. PENDAHULUAN

Teknologi Informasi adalah suatu teknologi yang digunakan untuk mengolah data, termasuk memproses, mendapatkan, menyusun, menyimpan, memanipulasi data dalam berbagai cara untuk menghasilkan informasi yang berkualitas, yaitu informasi yang relevan, akurat dan tepat waktu, yang digunakan untuk keperluan pribadi, bisnis, dan pemerintahan.

Peningkatan permintaan pada situs, menyebabkan *web server* sibuk menjawab permintaan klien dan terkadang pula web *server* mengalami kegagalan jika terlalu banyak permintaan sehingga *web server* tersebut tidak dapat menanganinya.

Teknologi *web server* dengan menggunakan *server* tunggal ternyata masih belum bisa memenuhi kebutuhan jaringan dengan maksimal. Ketika permintaan dari klien meningkat secara drastis, server tunggal tersebut tidak mampu menangani permintaan dari klien, sehingga yang terjadi web server tersebut menjadi *overload* dan *crash*. Masalah seperti inilah yang akhirnya menimbulkan suatu ide bagaimana membuat suatu jaringan yang *scalable*, *high-available* dan memiliki kemampuan *failover* ketika terdapat *server* yang mati dan akhirnya ditemukan teknologi *load balancing.*

Tugas akhir ini bertujuan untuk membangun sebuah rancangan layanan server yang saling terhubung dengan penyeimbang beban dan menganalisa performanya.

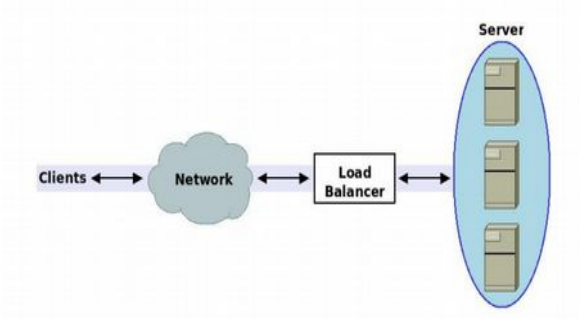
Perancangan basidata pada tugas akhir ini menggunakan MySQL yang berjalan di atas sistem operasi Linux Ubuntu 14.04 menggunakan *VirtualBox* dan tidak diimplentasikan pada mesin fisik. Tugas Akhir ini tidak membahas tentang *hacking* dan sistem keamanan pada sisi server dan klien. Tugas akhir ini menggunakan mesin virtual sebanyak dua buah server penyeimbang beban, dua buah server aplikasi dan dua buah server basisdata.

Studi referensi diambil dari penelitian Analisis Perbandingan Unjuk Kerja Sistem Penyeimbang Beban *Web Server* Dengan HAproxy Dan Pound Links oleh Dite Ardhian. Penelitian tersebut bertujuan untuk perbandingan unjuk kerja sistem penyeimbang beban *web server* dengan Haproxy dan Pound Links. Hasil penelitian tersebut menunjukan sistem penyeimbang beban menggunakan Haproxy memiliki keunggulan dibanding Pound Links dari parameter waktu respon 70% lebih cepat, *throughput* 20% lebih besar dan akses halaman per menit 60% lebih banyak.

II. DASAR TEORI

Penyeimbang Beban (*Load Balancing*) adalah suatu proses dan teknologi yang mendistribusikan trafik situs diantara beberapa *server* dengan menggunakan perangkat berbasis jaringan. Sesuai kriteria dasar penyeimbang beban proses ini mampu mengurangi beban kerja setiap *server* sehingga tidak ada *server* yang kelebihan beban, memungkinkan *server* untuk menggunakan *bandwidth* yang tersedia secara lebih efektif. Penyeimbang beban dapat diimplementasikan

dengan menggunakan perangkat keras, perangkat lunak, atau gabungan keduanya.[2][3]



Gambar 1 Skema Penyeimbang Beban[5]

HAproxy adalah singkatan dari *High Availability Proxy* produk *opensource* yg mendukung keperluan *load balancer* untuk TCP/HTTP. HAproxy mendistribusikan beban kerja pada satu set *server* untuk memaksimalkan kinerja dan mengoptimalkan sumber daya. Beban aplikasi pada *front-end* yang sangat bergantung pada *back-end* database dapat dengan mudah didistribusikan walau dengan banyaknya koneksi yang berjalan secara bersamaan. Semua klien akan terhubung kebagian *server* penyeimbang beban dan proxy akan meneruskan ke salah satu *server* basisdata yang tersedia berdasarkan skema *load-balancing* yang digunakan.[8]

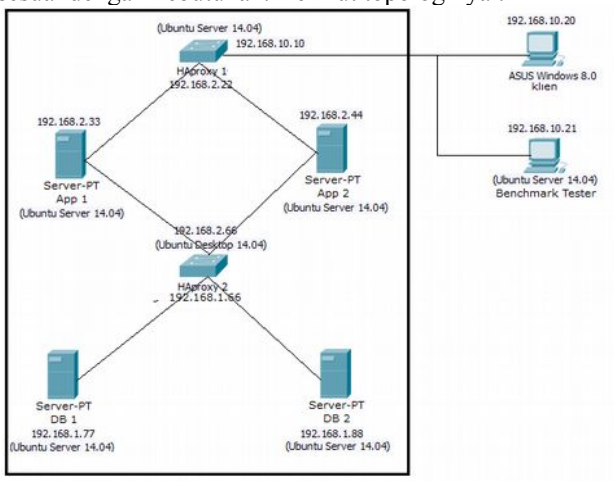
Web server adalah perangkat lunak yang menjadi tulang belakang dari *world wide web* (www). *Web server* menunggu permintaan dari klien yang menggunkan *browser* seperti Netscape Navigator, Internet Explorer, Mozilla Firefox, dan *browser* lainnya. Jika ada permintaan dari *browser*, maka *web server* akan memproses permintaan itu kemudian memberikan hasil prosesnya berupa data yang diinginkan kembali ke *browser*. Data ini mempunyai format standar, disebut dengan format SGML *(standart general markup language)*. Data yang berupa format ini kemudian akan ditampilkan oleh *browser* sesuai dengan kemampuan *browser.* Contohnya, bila data yang dikirim berupa gambar, *browser* yang hanya mampu menampilkan teks (misalnya *lynx)* tidak akan mampu menampilkan gambar tersebut, dan jika ada akan menampilkan alternatifnya saja.[1]

Replikasi adalah duplikasi data dari sumber basisdata, dalam hal ini disebut *master* ke basis data lainnya atau juga disebut *slave* dalam suatu jaringan. Replikasi di MySQL merupakan proses dinamis dari sinkronisasi data antara *server* (*master*) basis data utama dengan satu atau lebih sekunder (*slave*) *server* basis data secara *real-time*.[1][6]

Httperf adalah program untuk mengukur kinerja atau performansi dari web server yang dibuat oleh David Mosberger dari HP Labs. Httperf menyediakan fitur yang fleksibel dalam pembuatan beban kerja sesuai dengan parameter yang diberikan padanya. Httperf merupakan sebuah tool untuk sistem operasi turunan UNIX. Httperf dapat membangkitkan sejumlah beban paket dan mendukung HTTP/1.0 dan HTTP/1.1. Httperf dalam mengukur performansi sebuah *web server* adalah dengan mengirimkan permintaan atau paket ke server dengan rata tertentu dan mengukur waktu balasan yang diterima. [4][7]

III. PERANCANGAN SISTEM

Topologi jaringan merupakan hal yang menjelaskan hubungan geometris antara unsur-unsur dasar penyusun jaringan. Topologi jaringan sistem dibuat dengan menggunakan aplikasi *Packet Tracer* yang disusun sesuai dengan kebutuhan. Berikut topologinya :



Gambar 2 Topologi Sistem

Berdasarkan topologi jaringan pada Gambar 2 menjelaskan terdapat dua server penyeimbang beban diberi nama Hprx1 dan Hprx2, dua server aplikasi diberi nama App1 dan App2 , dua server basisdata diberi nama db1 dan db2 dan 2 klien diberi nama klien dan *benchmark tester*. Server virtualisasi ini juga memiliki dua bagian atau kelompok server yaitu, *DatabaseServer* dan *AppServer*

Perangkat lunak untuk penyeimbang beban menggunakan HAproxy versi 1.4. Konfigurasi haproxy dilakukan untuk mendeskripsikan sever yang ditangani, mode dan algoritma yang digunakan, port yang diijinkan, siapa yang menjadi front-end dan back-end dan konfigurasi lainnya. HAproxy dapat diinstal dengan menuliskan perintah :# `apt-get install haproxy`. Setelah instalasi selesai, edit file pada `etc/default/haproxy` ubah nilai `ENABLED=0` menjadi `ENABLED=1`. Konfigurasi utama haproxy terletak pada `/etc/haproxy/haproxy.cfg`.

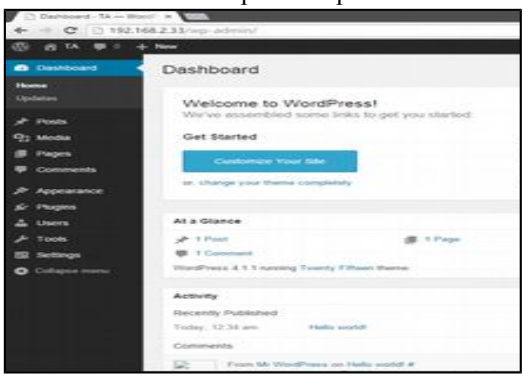
Server aplikasi diinstall aplikasi Wordpress, LAMP *Server* dan GlusterFS. Apache *web server* bertugas menerima dan membalas permintaan situs yang datang dari klien di *web server.* PHP5 berfungsi mendukung

bahasa pemrograman PHP pada *web server*. MySQL *server* sebagai basis data berfungsi sebagai syarat instalasi aplikasi Wordpress. GlusterFS digunakan untuk mereplikasi data Data yang direplikasi berada pada direktori root Apache yaitu `/var/www/html`. Pada direktori tersebut tersimpan beberapa file php yang digunakan oleh aplikasi *Wordpress*.

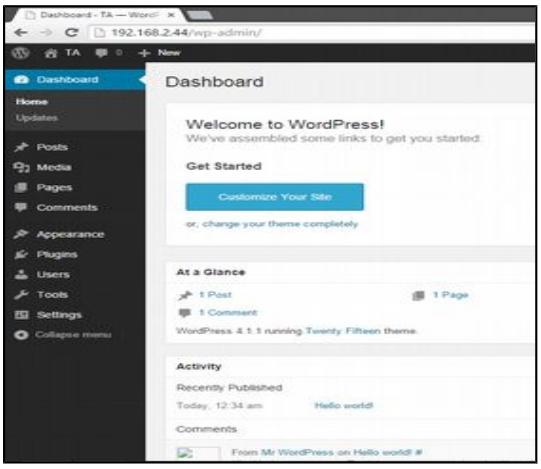
Server basisdata digunakan untuk menyimpan data klien atau data yang digunakan oleh wordpress seperti *user name* dan *password*. Server basidata menggunakan MySQL-server versi 5.5 yang direplikasi dua arah *master to master*. Konfigurasi dilakukan dikedua basisdata. Replikasi basisdata pada sistem yang dibuat disini membutuhkan konfigurasi *file* pada `/etc/mysql/my.cnf`. *File* tersebut dikonfigurasi untuk mendeskripsikan parameter yang digunakan dalam replikasi. Parameter yang hanya digunakan untuk dikonfigurasi yaitu *server-id*, *log_bin*, dan *binlog_do_db*.

IV. ANALISIS DAN PENGUJIAN

Pengujian dilakukan dengan cara klien mengakses ke server aplikasi dan melakukan sejumlah permintaan. Klien melakukan *login* sebagai admin pada kedua *server* aplikasi langsung dengan menuliskan 192.168.2.33/wp-admin dan 192.168.2.44/wp-admin pada browser.



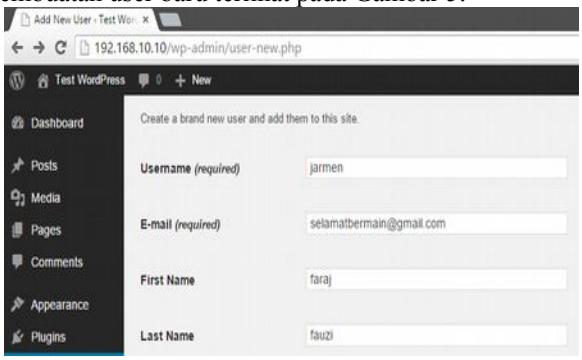
Gambar 3 Klien berhasil login pada server app1



Gambar 4 Klien berhasil login pada server app2

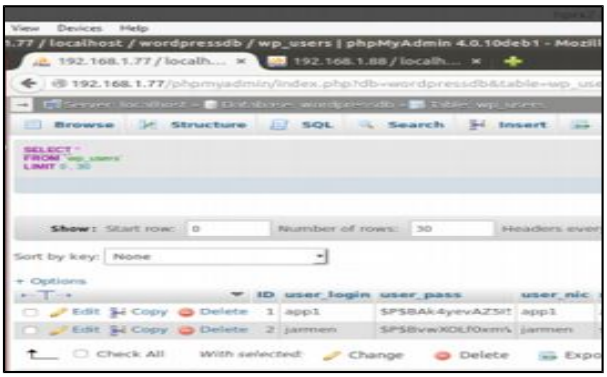
Klien sudah berhasil login pada kedua server aplikasi app1 berada pada alamat 192.168.2.33 yang ditunjukan oleh Gambar 3 dan app2 berada pada alamat 192.168.2.44 yang ditunjukan oleh Gambar 4. Pengujian ini menunjukan bahwa basisdata mysql pada server aplikasi (*localhost*) sudah dapat terkoneksi oleh aplikasi wordpress.

Pengujian replikasi dilakukan dengan membuat user baru yang disimpan dikedua basisdata secara otomatis. Basisdata yang direplikasi adalah wordpressdb. Pembuatan user baru terlihat pada Gambar 5.

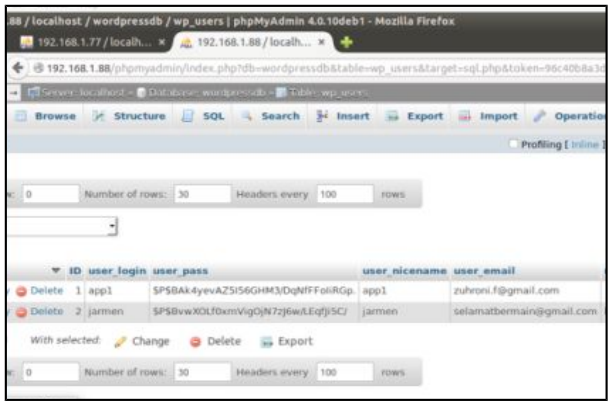


Gambar 5 Pembuatan user baru

Replikasi *master to master* memungkinkan perubahan yang dilakukan pada db1 pada wordpress juga berdampak pada db2 dan sebaliknya.

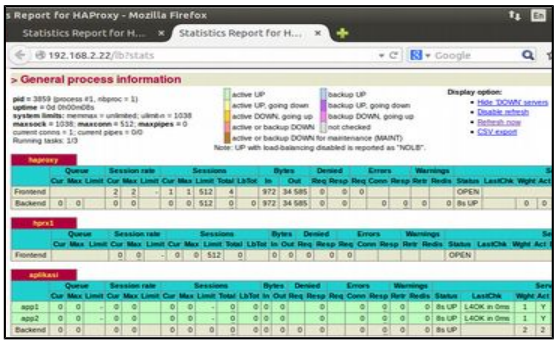


Gambar 6 User disimpan pada db1

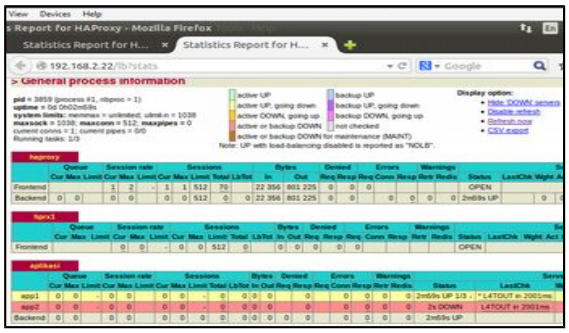


Gambar 7 User disimpan pada db1

Gambar 6 dan Gambar 7 menunjukan pembuatan user baru pada bernama *jarmen* sudah berhasil dibuat pada kedua server basisdata.

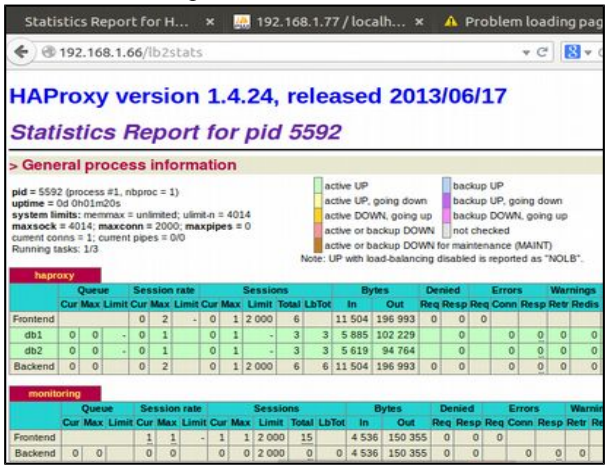


Gambar 8 Server aplikasi hidup



Gambar 9 Server aplikasi mati

Gambar 8 dan Gambar 9 menunjukan tampilan monitoring statistik server pada HAproxy. Halaman web statistik HAproxy tersebut dapat dilihat dengan menuliskan 192.168.1.22/lb?stats sesuai dengan konfigurasi yang dilakukan pada bab sebelumnya. Gambar 8 menunjukan server aplikasi yang berada pada kondisi hidup (*up*) ditandai dengan warna hijau sedangkan Gambar 9 menunjukan status server aplikasi mati (*down*) ditandai dengan warna merah dan kuning saat server dalam proses mati



Gambar 10 Statistik hprx2 saat klien melakukan login

Gambar 10 menunjukan hasil pada saat klien melakukan *login* wordpress sebagai admin. Proses login ini membutuhkan akses ke server basisdata untuk memeriksa apakah nama pengguna dan *password* yang diltulis oleh klien sesuai atau tidak dengan data yang berada pada server basisdata. Proses *login* tersebut diulangi sebanyak 6 kali hasilnya server penyeimbang beban berhasil meneruskan permintaan ke *server* db1 dan db2 sebanyak 3 kali secara bergantian.

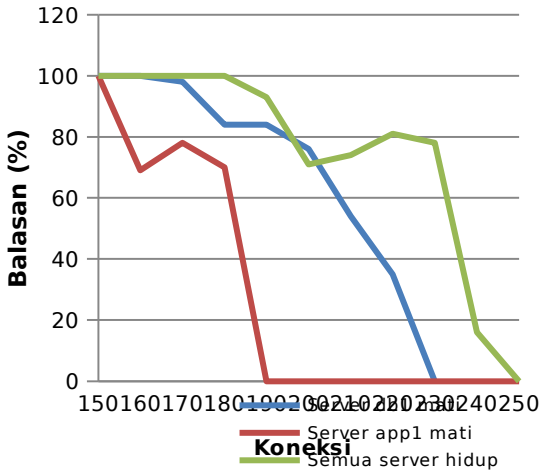
Pengujian penyeimbang beban menggunakan Httperf dapat dilakukan pada klien yang menggunakan sistem operasi Linux. Koneksi yang diberikan pada server ditingkatkan secara bertahap. Beban yang diberikan mulai dari 150 sampai dengan 250 koneksi dan menciptakan 100 beban permintaan. Sekenario pengujian dilakukan dengan tiga kondisi. Kondisi pertama saat

semua server hidup. Kondisi kedua saat salah satu server aplikasi mati. Kondisi ketiga saat salah satu server basisdata mati. Hasil seluruh pengujian terlampir pada Tabel 4.1, Tabel 4.2 dan Tabel 4.3.

Hasil dari keseluruhan pengujian saat semua kondisi server dalam keadaan hidup. Server secara optimal berada pada pengujian ke 4 dimana koneksi yang diberikan sejumlah 180, hasilnya mencapai *troughput* maksimal 207 KB/s tanpa adanya permintaan yang error. Server yang dirancang tidak mampu melayani permintaan melebihi dari 250 koneksi.pengujian ini dilakukan saat semua kondisi server dalam keadaan hidup

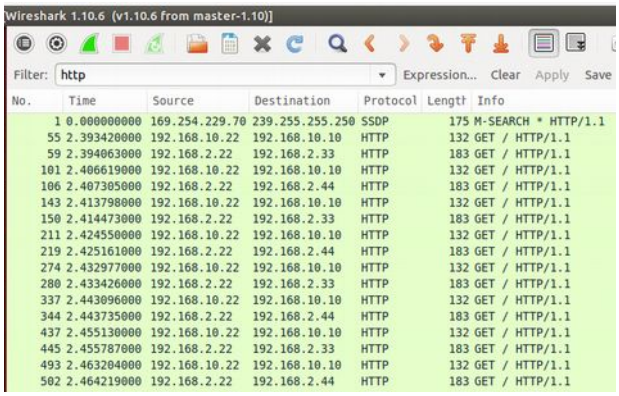
Hasil pengujian saat server app1 mati, server yang dirancang masih mampu melayani permintaan sampai 180 koneksi. Server mengalami kegagalan saat koneksi yang diberikan sebanyak 190. Kemampuan optimal saat salah satu server mati berada di 150 koneksi yang menghasilkan *troughput* sebesar 218.7 KB/s tanpa adanya koneksi yang *error*.

Hasil pengujian saat server app1 mati, server yang dirancang masih mampu melayani permintaan sampai 220 koneksi. Server mengalami kegagalan saat koneksi yang diberikan sebanyak 230. Kemampuan optimal saat salah satu server mati berada diantara 150 - 160 koneksi yang menghasilkan *troughput* sebesar 202.2 - 205.8 KB/s tanpa adanya permintaan yang *error*.



Gambar 11 Grafik hasil koneksi dan balasan

Gambar 11 adalah grafik hasil dari pengujian koneksi yang diberikan terhadap server dan persentase balasan yang diterima oleh klien menggunakan httperf. Pengujian dilakukan dengan tiga kondisi dan dibandingkan. Kondisi pertama adalah saat semua server dalam kondisi hidup. Hasil pengujian pada kondisi pertama menunjukan server mampu melayani permintaan mencapai 240 koneksi dan kemampuan optimal berada pada 180 koneksi. Kondisi kedua adalah server aplikasi app1 mati. Hasil dari kondisi kedua menunjukan server mengalami kegagalan saat koneksi yang diberikan sebanyak 190 koneksi. Kondisi terakhir adalah saat sever basisdata db1 mati. Server yang dirancang mampu melayani hingga 220 koneksi dan dapat melayani permintaan secara sempurna hingga 160 koneksi.



Gambar 12 Monitoring menggunakan *wireshark*

Gambar 1 menunjukan hasil monitoring menggunakan *wireshark* pada saat pengujian dilakukan. Komputer penguji berada pada alamat 192.168.10.22 (*source*) dan server yang dituju beralamat pada 192.168.10.10 (*destination*). Hasil monitoring menunjukan bahwa protokol yang digunakan oleh httperf dalam mengirimkan paket data adalah HTTP.

V. PENUTUP

Berdasarkan hasil pengujian dan analisis pada BAB 4, dapat diambil beberapa kesimpulan. Sistem yang dirancang mampu melakukan pembaharuan data secara *realtime* baik pada server aplikasi maupun basisdata. Hasil optimal saat kondisi server semua hidup adalah 180 koneksi, salah satu server aplikasi mati 150 koneksi dan saat salah satu basisdata mati adalah 230 koneksi. Hasil pengujian menunjukan kemampuan maksimal server dalam melayani saat kondisi server hidup berjumlah 220 koneksi, kondisi salah satu server aplikasi berjumlah 180 koneksi dan salah satu server basisdata mati berjumlah 220 koneksi. Sistem yaang dirancang dapat melakukan monitoring untuk pengecekan status pada server setiap 5 detik. Hasil monitoring menunjukan Httperf menggunakan protokol HTTP dalam mengirimkan paket data pada saat pengujian.

Beberapa saran yang dapat diberikan sehubungan penelitian ini yaitu, rancangan dapat diimplentasikan pada perangkat fisik dengan menambahkan kapasistas memori dan prosesor agar kemampuan server lebih optimal dan mudah dalam menganalisa. Penambahan server aplikasi yang lebih banyak untuk memaksimalkan fitur yang ada pada penyeimbang beban HAproxy dan Untuk mengetahui karakteristik kinerja sistem

penyeimbang beban. Perlu adanya penelitian lebih lanjut mengenai algoritma penjadwalan yang digunakan dalam load balancing untuk mengoptimalkan kinerja sistem.

DAFTAR PUSTAKA

[1] Ardhian, D. (2013). *Analisis Perbandingan Unjuk Kerja Sistem Penyeimbang Beban Web Server Dengan HAproxy dan Pound Links*. Semarang: Skripsi S-1 Universitas Diponegoro.

[2] Asbin, M. (2010). *Implementasi Load Balancing Pada Web Server*. Medan: Skripsi S-1 Universitas Sumatra Utara.

[3] Bourke, T. L. (2001). *Server LoadBalancing*. United States of Amecira: O'Reilly.

[4] David Mosberger. (1998). *Httperf: a Tool for Measuring Web server Performance*. California, Palo Alto.

[5] Hagen, W. v. (2010). *Ubuntu Linux Bible : Featuring Ubuntu 10.04 LTS*. Indiapolis: Wiley Publishing Inc.

[6] Huda, M. (2004). *Membuat Aplikasi Database Dengan Java, MySQL dan NetBeans*. Jakarta: Bunafit Komputer.

[7] Theodore, B. (2007). *httperf: Web Workload Generator Quickstart Guide*.

[8] --,Haproxy Documentation, http://cbonte. github .io/haproxy-dconv/configuration-1.html, 10 April 2001

[9] Purnoma. (2010). *Membangun Virtual PC dengan VirtualBox*. Yogyakarta: Penerbit ANDI.

[10] Suyadi. (2011). *Membuat Media Penyimpanan Terdistribusi Menggunakan GlusterFS pada Debian Squeeze*. Surakarta: Skripsi S-1 Universitas Muhammadiyah.

[11] Wahana, K. (2001). *Mari Mengenal Linux*. Semarang: Penerbit ANDI.